DENDY
ПОД МИКРОСКОПОМ

Архитектура

Система команд

Программирование

Анализ программ

Екатеринбург 1995

От автора

У Вас есть DENDY? А что Вы о ней знаете? Ничего не знаете и не хотите знать…

Ну что ж, эта брошюра не для Вас. Она для тех, кому в детстве интереснее было заглянуть в заводную машинку, чем катать её по полу.

Если же Вы хотите разобраться, что представляет из себя ругаемая на все лады, но буквально заполонившая наше Отечество 8-битная видеоприставка, научиться изучать её программы, а может быть и писать свои, подключить к ней магнитофон или дисковод, научиться адаптировать игры из картриджей на ленту или дискету - эта брошюра поможет Вам сделать первый шаг.

К сожалению фирмы, выпускающие видеоприставки, почему-то не торопятся опубликовать информацию о них. Я хочу восполнить эту недоработку. Надеюсь, что информация, которую я по крохам собирал из разных источников и добыл с осциллографом в руках, пригодится широким массам радиолюбителей и хакеров.

Это неофициальное издание и поэтому текст не подвергался корректуре и редакторской правке. Все замечания, пожелания и уточнения будут приняты с благодарностью.

Екатеринбург тел. 3432-340-124 Веремеенко С. Л.
(ZX-EXPERT)

Введение

Для начала давайте договоримся, что я любую 8-битную видеоприставку буду называть "DENDY". Несмотря на уверения фирм-изготовителей, все они одинаковы. Отличия носят непринципиальный характер, да иначе и быть не может. Ведь любая более-менее серьезная модификация неминуемо отразится на совместимости. Несколько выпадает из этого ряда "компьютер" SONIC. У него непонятно зачем расширена память и подключена клавиатура, но вообще-то это типичная DENDY.

Я не могу гарантировать полноту и абсолютную точность информации, приведенной в этой брошюре, учитывая источники, которыми я пользовался. Во всех случаях, когда достоверность не может быть гарантирована, я буду специально отмечать это.

Текст составлялся с таким расчетом, чтобы что-то понял даже тот, кто о компьютерах ничего не знает, поэтому не обижайтесь, если Вам покажется, что я разжевываю очевидные вещи. В то же время у меня нет возможности превращать эту брошюру в учебник по электронике и программированию. Она рассчитана в основном на уровень, ненамного превышающий Вузовский, т.е. уровень среднего радиолюбителя.

В этой брошюре приводится информация по архитектуре DENDY, описание микропроцессора и его системы команд, рассмотрены разновидности картриджей, приведена методика съема информации с картриджей, даны рекомендации по взлому, адаптации и, в какой-то мере, написанию программ для DENDY.

Уже подготовлены к изданию и выйдут в ближайшее время брошюры с описанием практических конструкций дополнительным внешних устройств для DENDY - магнитофонного адаптера и дисковода. Кроме описания самих конструкций в них будут приведены методики переноса игр с картриджей на ленту и дискеты. Разработаны программы, позволяющие русифицировать игры и устанавливать в них "бессмертие", "неуязвимость" и т.д. Все "хитрое" программное обеспечение разработано для компьютера ZX-SPECTRUM 128 кб.

Архитектура DENDY

Как и в любой микро-ЭВМ, наибольшее влияние на выбор её архитектуры оказал выбор микропроцессора. Мне трудно судить, почему разработчики DENDY остановили свой выбор на далеко не самом совершенном и удачном микропроцессоре серии 65XX. По всей видимости, основными были соображения облегчения адаптации (это иностранное слово, Вам не понять. По-русски "сдирания") программного обеспечения APPLE-2, COMMODORE, ATARY. Но что сделано, то сделано.

Этот микропроцессор имеет ряд, мягко выражаясь, особенностей, довольно сильно отличающих его от других, более удачных. Он имеет фиксированную область стека по адресам 0100h-01FFh и фиксированные вектора холодного старта и прерываний по адресам FFFAh-FFFFh, что в совокупности с отсутствием у него поля портов ввода/вывода однозначно определяет разорванность адресного пространства в любой системе, основой которой он является. И DENDY в этом смысле не является исключением.

На рис.1 приведена карта распределения адресного пространства. По адресам 0000h-07FFh размещено встроенное ОЗУ объемом 2 кб. Вследствие неполной его адресации оно отображается также на адреса 0800h-1FFFh. С адреса 2000h начинается область портов ввода/вывода. Адреса 2000h-2007h заняты видеопроцессором, а 4000h и далее джойстиками и световым пистолетом, если судить по расковырянным мной играм, хотя, как ни странно, аппаратной связи этих адресов с джойстиками нет.

Распределение памяти

Картридж	FFFFh 8000h
Резерв	6000h
Джойстики	4000h
Видеопроцессор	2000h
Внутреннее ОЗУ	0000h

Рис.1

Этот вопрос требует дополнительного изучения. Возможно это особенность - именно 6527 в отличие от 6502.

Начиная с адреса 8000h и до конца идет адресное пространство картриджа. Т.е. непосредственно без страничной адресации емкость ЗУ картриджа может составить 32 кб.

Кстати об одном недоразумении. "За бугром" принято считать ёмкость ПЗУ в битах. Поэтому 1М означает не 1 мегабайт, как многие думают, а всего 128 кб.

Видеопроцессор имеет собственное адресное пространство в 16 кб. ВидеоПЗУ картриджа размещается по адресам 0000h-1FFFh. С адреса 2000h размещено встроенное ОЗУ емкостью 2 кб, также как и встроенное ОЗУ CPU, адресованное не полностью, но его отображение на старшие адреса имеет некоторые особенности, которые мы рассмотрим, когда дело дойдет до описания картриджей. Видеопроцессор может генерировать пребывания для CPU, причем они являются немаскируемыми. Вход маскируемых прерываний выведен на разъем картриджа, но используется крайне редко.

Центральный процессор

Микропроцессор 6527 или его аналоги, используемый в DENDY, как уже указывалось, относится к семейству 65XX. Отличительными его особенностями является наличие встроенного музпроцессора и нескольких дополнительных линий, используемых как одноразрядные порты ввода/вывода.

По всей видимости, для обслуживания этих, дополнительных по отношению к 6502, возможностей введены дополнительные команды. Вероятность получения исчерпывающей информации о нем от фирмы изготовителя близка к нулю, а у меня до музыки руки пока не дошли.

Поэтому я приведу здесь систему команд 6502, которая является подмножеством системы 6527. Это утверждение проверено на практике. Я написал для DENDY достаточно много программ, пользуясь кроссассемблером настроенным на систему команд 6502 и ни разу не столкнулся с чем-то непонятным. Даже количество тактов микропроцессора совпадает для всех команд. Поэтому можно с большой долей уверенности утверждать, что описание микропроцессора 6502 вполне подойдет для практической работы с 6527.

Микропроцессор 6502 существенно отличается от других 8-битных процессоров.

Он имеет всего три 8-разрядных регистра: регистр А-аккумулятор, регистры X и Y, используемые как индексные или как дополнительные аккумуляторы. К сожалению, использованию их как полноценных индексных препятствует их малая разрядность, а как дополнительные аккумуляторы они могут быть использованы только с большими

ограничениями.

Кроме того, есть 9-разрядный регистр стека (S), старший разряд которого всегда 1, и 8-разрядный регистр флагов (P). И это все. Тому, кто привык к роскоши Z-80 или 8088, программировать будет тоскливо.

В области старших адресов размещается три вектора:

Немаскируемое прерывание NMI - FFFAh

Холодный старт RESET - FFFCh

Маскируемое прерывание IRQ - FFFEh

Особенностью, которую обязательно нужно учитывать при программировании, является автоматическое занесение в стек регистра флагов при прерывании.

Формат регистра флагов (слово состояния):

```
0 разряд - C        ; перенос. 1 при сложении, 0 при вычитании
1 разряд - Z        ; ZERO. 1, если результат равен 0
2 разряд - I        ; INT. 1 - прерывание запрещено
3 разряд - D        ; DECIM. 0 - двоичный счет, 1 - десятичный
4 разряд - B        ; BRK. 1 при выполнении команды BRK
5 разряд - ?        ; всегда 1
6 разряд - V        ; арифметическое переполнение
7 разряд - N        ; SIGN Знак числа, устанавливается по D7
```

Крайняя скудность системы команд 6502 в какой-то мере компенсируется большим количеством способов адресации.

Методы адресации.

1 ACC - аккумуляторная, операнд в аккумуляторе

2 IMPL - неявная

3 IMM - непосредственная, операнд - аргумент команды

4 ABS - прямая длинная, операнд в ячейке, заданной аргументом - полным адресом

5 ZP - прямая короткая, операнд в ячейке 0 страницы. Адрес - только младший байт, старший всегда 0

6 ZP,X - индексная короткая, операнд в ячейке по адресу аргумент команды +X. Только в пределах 0 страницы, иначе ошибка!

7 ZP,Y - аналогично, но по Y.

8 ABS,X - индексная длинная. Адрес полный, ограничений нет

9 ABS,Y - аналогично, но по Y

10 REL - относительная для коротких условных переходов

11 IND,X - идексно-косвенная по X. Адрес исполнительного адреса вычисляется как сумма аргумента и X. Перенос не учитывается.

12 IND,Y - косвенно-индексная по Y. Исполнительный адрес вычисляется как сумма Y и содержимого пары ячеек, адресуемых аргументом.

13 IND - косвенная длинная. Адрес располагается по адресу, определяемому аргументом. Аргумент - полный адрес.

Ну и как Вам это нравится? Впрочем, на самом деле ничего страшного, навык использовать все эти навороты появляется довольно быстро, особенно когда нет другого выхода.

Отмечу некоторые особенности. Адреса располагаются в памяти по принципу "младший байт в младшей ячейке", так же как и I8080 и Z-80. Десятичный режим определяется только для команд ADC и SBC, адреса вычисляются по-прежнему в двоичной системе счисления. В отличие от I8080 перенос при вычитании устанавливает флаг C в 0. Адрес короткого условного перехода вычисляется в дополнительном двоичном коде от команды следующей за переходом, так же как и у Z-80. Нужно быть трижды внимательным с флагами, т.к. их может изменить даже команда загрузки регистра (предел маразма!).

Короткого безусловного перехода нет, как, впрочем, и длинных условных. Также нет условных вызовов и выходов из подпрограмм. Команда BIT, не меняя содержимого аккумулятора, устанавливает флаг Z по результату операции И над аккумулятором и операндом, а флаги N и V копирует из разрядов D7 и D6 операнда. Несмотря на её необычность, очень полезная и приятная команда.

По сбросу содержимое регистров, флагов и режима неопределено, обо всем должен позаботиться программист.

Стек располагается в первой странице и, если вы не хотите приключений, ничего в ней не располагайте. Нулевая страница особенная. Вся короткая и индексная адресация идет через нее, поэтому там следует располагать только адреса и переменные, которые могут трактоваться как адреса. Точно сказать не могу, но очень похоже, что адреса 0000h - 000Fh и 00A0h - 00FFh микропроцессор 6527 использует для внутренних нужд. Смело можете использовать область 0010h - 009Fh.

Команда BRK производит тот же эффект, что и маскируемое прерывание, но кроме того, устанавливает флаг B. Регистр указатель стека прямо не загружается. Загрузить или считать его можно только через регистр X. Команды ADD нет, во всех случаях приходится пользоваться командой ADC. Не забывайте в необходимых случаях сбрасывать флаг переноса.

Аппаратное быстродействие 6527 в 2 - 3 раза больше, чем например у Z-80, при равных тактовых частотах, но поскольку одну команду Z-80 приходится заменять тремя - пятью 6527, реальное быстродействие 6527 по крайней мере не больше, чем у Z-80. Во всяком случае, пересылки больших массивов происходят заметно медленнее.

Особенности системы команд 6527 заставляют каждую процедуру оформлять как подпрограмму. Тексты игр состоят, в основном, из сплошных вызовов мелких, по 5-10 команд, подпрограмм.

Система команд

Мнемоника	Функция	Метод адресации	Код HEX	К-во байт	Флаги
ADC	A=A+M+C	IMM ZP ZP, X ABS ABS, X ABS, Y (IND, X) (IND), Y	69h 65h 75h 6Dh 7Dh 79h 61h 71h	2 2 2 3 3 3 2 2	N, V, Z, C
AND	A=A and M	IMM ZP ZP, X ABS ABS, X ABS, Y (IND, X) (IND), Y	29h 25h 35h 2Dh 3Dh 39h 21h 31h	2 2 2 3 3 3 2 2	N, Z
ASL	арифметический сдвиг влево D0=0, C=D7	ACC ZP ZP, X ABS ABS, X	0Ah 06h 16h 0Eh 1Eh	1 2 2 3 3	Z, C
BCC	IF C=0	REL	90h	2	
BCS	IF C=1	REL	B0h	2	

Мнемоника	Функция	Метод адресации	Код HEX	К-во байт	Флаги
BEQ	IF Z=1	REL	F0h	2	
BIT	A and M	ZP ABS	24h 2Ch	2 3	N=M7, V=M, Z
BMI	IF N=1	REL	30h	2	
BNE	IF Z=0	REL	D0h	2	
BPL	IF N=0	REL	10h	2	
BRK	программное прерывание	IMPL	00h	1	
BVC	IF V=0	REL	50h	2	
BVS	IF V=1	REL	70h	2	
CLC CLD CLI CLV	C=0 D=0 I=0 V=0	IMPL IMPL IMPL IMPL	18h D8h 58h B9h	1 1 1 1	сброс в 0 выбранного флага
CMP	A-M сравнение	IMM ZP ZP, X ABS ABS, X ABS, Y (IND, X) (IND), Y	C9h C5h D5h CDh DDh D9h C1h D1h	2 2 2 3 3 3 2 2	N, Z, C
CPX	X-M сравнение	IMM ZP ABS	E0h E4h ECh	2 2 3	N, Z, C
CPY	Y-M сравнение	IMM ZP ABS	C0h C4h CCh	2 2 3	N, Z, C
DEC	M=M-1	ZP ZP, X ABS ABS, X	C6h D6h CEh DEh	2 2 3 3	N, Z
DEX	X=X-1	IMPL	CAh	1	N, Z
DEY	Y=Y-1	IMPL	88h	1	N, Z
INX	X=X+1	IMPL	E8h	1	N, Z
INY	Y=Y+1	IMPL	C8h	1	N, Z
EOR	A=A xor M исключающее или	IMM ZP ZP, X ABS ABS, X ABS, Y (IND, X) (IND), Y	49h 45h 55h 4Dh 5Dh 59h 41h 51h	2 2 2 3 3 3 2 2	N, Z
INC	M=M+1	ZP ZP, X ABS ABS, X	E6h F6h EEh FEh	2 2 3 3	N, Z
JMP	переход	ABS IND	4Ch 6Ch	3 3	

Мнемоника	Функция	Метод адресации	Код HEX	К-во байт	Флаги
JSR	обращение к подпрограмме	ABS	20h	3	
LDA	A=M	IMM ZP ZP, X ABS ABS, X ABS, Y (IND, X) (IND), Y	A9h A5h B5h ADh BDh B9h A1h B1h	2 2 2 3 3 3 2 2	N, Z обратите особое внимание!
NOP	нет операции	IMPL	EAh	1	
LDX	X=M	IMM ZP ZP, Y ABS ABS, Y	A2h A6h B6h AEh BEh	2 2 2 3 3	N, Z
LDY	Y=M	IMM ZP ZP, Y ABS ABS, Y	A0h A4h B4h ACh BCh	2 2 2 3 3	N, Z
LSR	арифметический сдвиг вправо D7=0, C=D0	ACC ZP ZP, X ABS ABS, X	4Ah 46h 56h 4Eh 5Eh	1 2 2 3 3	N=0, Z, C
OR A	A= A or M	IMM ZP ZP, X ABS ABS, X ABS, Y (IND, X) (IND), Y	09h 05h 15h 0Dh 1Dh 19h 01h 11h	2 2 2 3 3 3 2 2	N, Z
PHA PHP PLA PLP	A в стек P в стек A из стека P из стека	IMPL IMPL IMPL IMPL	48h 08h 68h 28h	1 1 1 1	 флаги из стека
ROL	циклический сдвиг влево	ACC ZP ZP, X ABS ABS, X	2Ah 26h 36h 2Eh 3Eh	1 2 2 3 3	N, Z, C
ROR	циклический сдвиг вправо	ACC ZP ZP, X ABS ABS, X	6Ah 26h 36h 2Eh 3Eh	1 2 2 3 3	N, Z, C
RTI	возврат из прерывания	IMPL	40h	1	из стека

Мнемоника	Функция	Метод адресации	Код HEX	К-во байт	Флаги
RTS	возврат	IMPL	60h	1	
SBC	A=A-M-C	IMM ZP ZP, X ABS ABS, X ABS, Y (IND, X) (IND), Y	E9h E5h F5h EDh FDh F9h E1h F1h	2 2 2 3 3 3 2 2	N, V, Z, C
STA	M=A	ZP ZP, X ABS ABS, X ABS, Y (IND, X) (IND), Y	85h 95h 8Dh 9Dh 99h 81h 91h	2 2 3 3 3 2 2	
STX	M=X	ZP ZP, Y ABS	86h 96h 8Eh	2 2 3	
STY	M=Y	ZP ZP, X ABS	84h 94h 8Ch	2 2 3	
SEC SED SEI TAX TAY TXA TYA TSX TXS	 запрет прерыв. X=A Y=A A=X A=Y X=указат.стека указ.стека=X	IMPL IMPL IMPL IMPL IMPL IMPL IMPL IMPL IMPL	38h F8h 78h AAh A8h 8Ah 98h BAh 9Ah	1 1 1 1 1 1 1 1 1	C=1 D=1 I=1 N, Z N, Z N, Z N, Z N, Z

В старых восьмикристальных DENDY устанавливался микропроцессор 6527, 6827, PH03 или TA03N. Они отличаются только фирмами - изготовителями. Индекс P или S показывает для какой телевизионной системы (PAL или SECAM) они предназначены.

Цоколёвка микропроцессора приведена на Рис.2.

Цоколёвка процессора 6527

Вывод	Сигнал	Вывод	Сигнал
1	Sound1	40	+5V
2	Sound1	39	Lood1
3	Reset	38	Lood2
4	A0	37	Lood3
5	A1	36	IMP 0
6	A2	35	IMP 1
7	A3	34	W/R
8	A4	33	NMI
9	A5	32	INT
10	A6	31	RDY
11	A7	30	TEST

12	A8	29	CLK
13	A9	28	D0
14	A10	27	D1
15	A11	26	D2
16	A12	25	D3
17	A13	24	D4
18	A14	23	D5
19	A15	22	D6
20	GND	21	D7

Рис.2

Рекомендации по программированию

Архитектура микропроцессора 6502 (6527) накладывает определённые ограничения на стиль программирования. Самое основное, что у этого микропроцессора нет 16 разрядной арифметики. Поэтому операции с адресами выглядят совершенно иначе, чем например у Z-80.

Рассмотрим простейший пример. Предположим, нам нужно инкрементировать некий указатель адресов. Такого рода задача встает очень часто при заполнении массива определенным числам или переносе из одной области памяти в другую. Для Z-80 просто называем регистр HL указателем, и команда выглядит так:

```
INC  HL   ; длина 1 байт
```

Для 6502 указатель придется организовать в памяти, т.к. 16 разрядного регистра нет, а регистры X и Y в пару не объединяются.

```
COUNT     EQU  80h        ; для примера адрес 80h
INCREM    CLC             ; сброс переноса
          LDA  #01h       ; величина инкремента
          ADC  COUNT
          STA  COUNT      ; младший байт указателя
          LDA  #00h       ; очистка аккумулятора
          ADC  COUNT+1
          STA  COUNT+1    ; старший байт указателя
          RTS             ; длина 14 байт!
```

Разумеется, этот пример несколько утрирован. Он просто доказывает, во что превратится программа, если механически переносить привычные приемы программирования на непривычный микропроцессор.

Чтобы не писать программ похожих на этот пример, нужно с самого начала отказаться от попыток применения опыта программирования, приобретенного на других типах микропроцессоров.

А все-таки как, же решить эту задачу?

Во-первых, если нет 16 разрядного счетчика, значит его нет. Вместо одного цикла с 16 разрядным счетчиком нужно использовать 2 с 8 разрядными. Во-вторых, нужно научиться использовать "хитрые" способы адресации, на которые этот микропроцессор горазд.

Пример подпрограммы заполняющей область памяти нулями:

```
COUNT     EQU  80h
          LDA  ADR1
          STA  COUNT+1
          LDX  ADR2
          LDA  #00h
```

```
            STA  COUNT
            TAY
L1          STA  (COUNT),Y
            INY
            BNE  L1
            INC  (COUNT+1)
            CPX  (COUNT+1)
            BNE  L1
            RTS  ; всего 23 байта
```

Аналогичная программа для Z-80:

```
            LD   HL,ADR
            LD   (HL),0
            LD   DE,ADR+1
            LD   BC,LENG
            LDIR
            RET  ; всего 14 байт
```

Как видите, разница в 1,5 раза.

Так как разрядность индексных регистров 8 бит, память естественным образом делится на 256 страниц по 256 байт. Оперируя со страницами, можно упрощать и сокращать размер программ. Не нужно стараться писать длинные подпрограммы. Оптимальный размер не более 20-30 байт. Иногда даже 5 байтную процедуру выгоднее оформить как подпрограмму. Счетчики и регистры временного хранения нужно организовывать в нулевой странице памяти. Она, в какой-то мере, заменяет недостающие регистры процессора. Обращения к упорядоченным таблицам лучше осуществлять через регистр Y, в то время как регистр X предоставляет уникальную возможность работы с неупорядоченными таблицами. Это, пожалуй, единственный случай, когда система команд 6502 оказывается эффективнее, чем у Z-80 и, похоже, даже I8080. Впрочем, это уже высший пилотаж.

Не поддавайтесь на искушение писать самомодифицирующиеся программы. 6502 прямо-таки провоцирует на это, но это почти всегда плохо кончается, особенно, если программа достаточно объемная. Ну и само собой, такая программа не будет работать в ПЗУ, а для DENDY это серьезное ограничение.

Видеопроцессор

С видеопроцессором дела обстоят несколько хуже. Мне удалось определить, что он управляется по 8 адресам. В DENDY для него задействованы адреса 2000h-2007h. Функции некоторых регистров удалось выяснить, анализируя игры. Практически все игрушки начинаются процедурой инициализации, которая выглядит приблизительно так:

```
            CLD             ; сброс десятичного режима
            SEI             ; запрет прерываний
            LDA  #00h       ; очистка аккумулятора
            STA  2000h
            STA  2001h
LOOP        LDA  2002h      ; ожидание установки
            BPL  LOOP       ; разряда D7 в 0
```

Кроме того, известно, что координаты позиции вывода на экран определяются регистром 2005h.

CPU через видеопроцессор может записывать и считывать информацию из памяти видеопроцессора. Адрес устанавливается путем записи двух байтов (вначале старший, потом младший) по адресу 2006h. Чтение или запись осуществляется путем передачи массива по адресу 2007h. Причем, при чтении первый считанный байт недостоверен. Только второй и последующие считываются из заданного адреса и могут быть использованы.

Видеопроцессор формирует полный видеосигнал полностью самостоятельно, не

нуждаясь в контроле CPU. Обращения CPU происходят только при необходимости сменить картинку. Но взаимодействие с видеопамятью идет непрерывно. Видеопроцессор может формировать немаскируемые прерывания для CPU. В некоторых случаях их генерация прекращается, но это никак не сказывается на картинке. Это пока всё, что я могу сообщить о видеопроцессоре. Надеюсь в ближайшем будущем узнать больше.

Может быть, Вам удастся выяснить что-то важное, прошу связаться со мной. В следующей брошюре о видеопроцессоре наверняка будет более полная информация.

Цоколёвка видеопроцессора приведена на рис. 3.

Цоколёвка видеопроцессора 6538

Вывод	Сигнал	Вывод	Сигнал
1	R/W	40	+5V
2	D0	39	ALE
3	D1	38	DV0
4	D2	37	DV1
5	D3	36	DV2
6	D4	35	DV3
7	D5	34	DV4
8	D6	33	DV5
9	D7	32	DV6
10	A2	31	DV7
11	A1	30	AV8
12	A0	29	AV9
13	CE	28	AV10
14	TEST0	27	AV11
15	TEST1	26	AV12
16	TEST2	25	AV13
17	TEST3	24	OE
18	NMI	23	V.R/W
19	CLK	22	RESET
20	GND	21	VIDEO

Рис.3

Разъём картриджа

В большинстве DENDY разъем картриджа предоставляет единственную возможность подключиться к ней, т.к. они однокристальные и добраться до внутренних узлов невозможно. Поэтому сигналам выведенным на разъём необходимо уделить самое пристальное внимание. Цоколёвка разъема приведена на рис. 4

Разъём картриджа

Сигнал	Сторона A	Сторона B	Сигнал
к CPU			
GND	1	1	+5V
A11	2	2	CLK
A10	3	3	A12
A9	4	4	A13
A8	5	5	A14
A7	6	6	D7
A6	7	7	D6
A5	8	8	D5

A4	9	9	D4
A3	10	10	D3
A2	11	11	D2
A1	12	12	D1
A0	13	13	D0
WR	14	14	CS
INT	15	15	SOUND
GND	16	16	SOUND
к видеопроцессору			
CS	17	17	WR
A10’	18	18	A13
A6	19	19	A13
A5	20	20	A7
A4	21	21	A8
A3	22	22	A9
A2	23	23	A10
A1	24	24	A11
A0	25	25	A12
D0	26	26	OE
D1	27	27	D7
D2	28	28	D6
D3	29	29	D5
+5V	30	30	D5

Рис.4

Разъем картриджа четко делится на две части. Контакты с 1 по 15 идут к CPU, а с 17 по 30 к видеопроцессору.

Рассмотрим вначале группу контактов, через которую CPU связан с картриджем. Мы видим обычную системную шину микропроцессора. Контакты B6-B13 образуют шину данных, A2-A11 и B3-B5 шину адреса. Адрес A15 на разъем не выведен, но сигнал выбора CS (B14) появляется только при A15=1.

Сигнал WR (A14) устанавливается в 0 в цикле записи и равен 1 в цикле считывания. На контакт A15 выведен вход маскируемого прерывания INT, а на контакт B2 синхросигнал микропроцессора CLK . Контакты A1 и A16 -земля, B1 - питание +5 вольт.

Сигналы CS, WR, INT инверсные, активен нулевой уровень, как и в большинстве микропроцессоров, а у сигнала CLK активен единичный уровень.

Все шины небуферизованы, поэтому допускается нагрузка не более 1 входа TTL. Сигнал WR шире сигнала CS, (смотри Рис. 5). Это очень удобно для подключения памяти, но вызовет затруднения, если Вы захотите подключить порт 580ВВ55. К сожалению, на разъем не выведен сигнал сброса.

При нажатии кнопки "Сброс" все шины переходят в третье (оборванное) состояние.

Если Вы захотите подключить к DENDY какое-то свое устройство, нужно учитывать, что хотя у большинства DENDY сетевые адаптеры на 650 мА, более чем на 50 мА нагружать ее источник +5в нельзя.

Это связано с тем, что ее внутренний стабилизатор слаб и еле тянет те 180-220 мА, которые потребляет сама DENDY. Однако вполне возможно к адаптеру подключить дополнительный стабилизатор, например типа 142ЕН5А, с которого можно снять 200 - 300 мА без вреда для DENDY.

Видеопроцессор имеет шину очень похожую на шину CPU. Так же есть шина данных (A26-A29, B27-B30), шина адреса (A19-A25 и B20-B25), CS (B26) и WR (B17). Кроме того, есть линия A10', назначение которой рассмотрим чуть позже, когда будем разглядывать картридж.

Контакты B15, B16 и B18, B19 на большинстве картриджей замкнуты между собой. Через B15, B16 проходит сигнал звукового сопровождения. Блокировка звука, возможно, предусмотрена чтобы избежать "рычания" телевизора, если кому-то взбредет в голову включить DENDY без картриджа.

Функция B18, B19 сложнее. Если для работы программы достаточно внутренней видеопамяти, то, изменив эту перемычку, можно подключить встроенное ОЗУ на адреса видеопроцессора 0000h-07FFh. В этом случае видеоПЗУ в картридже может отсутствовать. Правда, мне таких картриджей пока не попадалось.

Временная диаграмма

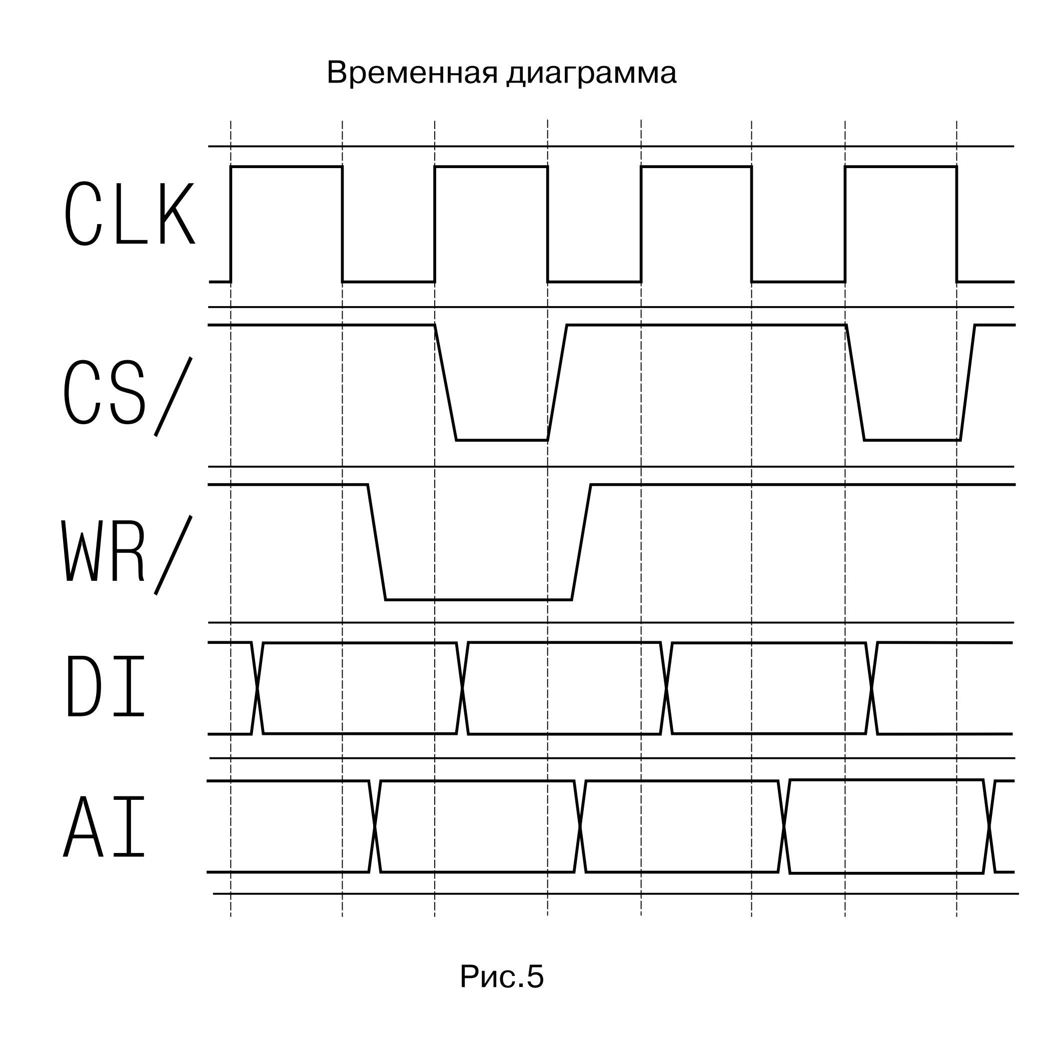


Рис.5

Устройство картриджа

Видимо Вам уже приходилось открывать картриджи для DENDY.

В картридже установлена небольшая печатная плата с торцевым печатным разъемом. На плате установлено от 2 до 5 - 6 микросхем, часто бескорпусных, залитых компаундом. С корпусными микросхемами более-менее ясно, но как определить тип и функции бескорпусных?

Большинство бескорпусных микросхем представляет собой ПЗУ емкостью от 8 до 128 кВ. Иногда возле такой микросхемы есть надписи типа 64к, 256к, 1М. Необходимо помнить, что ёмкость приведена в битах. Чтобы определить ёмкость в килобайтах, это число нужно разделить на 8. У микросхем ПЗУ, как правило, 20 выводов. Иногда один из выводов кристалла не выводится наружу и из под компаунда выходят 27 контактов. Дополнительным признаком ПЗУ можно считать то, что на него поданы все разряды шины данных и все или большая часть адресных линий. Цоколевка ПЗУ приведена на рис. 6. Вначале прозвонкой, находим выводы питания и земли, затем убеждаемся в соответствии линий шины данных и адреса.

Цоколевка ПЗУ 1М

Вывод	Сигнал	Вывод	Сигнал
1	A15	28	+5V
2	A12	27	A14
3	A7	26	A13
4	A6	25	A8
5	A5	24	A9
6	A4	23	A11
7	A3	22	A16
8	A2	21	A10
9	A1	20	OE
10	A0	19	D7
11	D0	18	D6
12	D1	17	D5
13	D2	16	D4
14	GND	15	D3

Рис.6

Кроме ПЗУ в состав картриджа может входить статическое КМОП ОЗУ емкостью 2, 8 или 32 кб. Такие микросхемы можно отличить по тому, что на них заводится сигнал WR, кроме того, они, как правило, корпусные.

Третья разновидность микросхем, применяемых в картриджах - заказные ПЛМ. Они всегда бескорпусные с количеством выводов от 20 до 40. Отличительной особенностью ПЛМ является нерегулярное ее подключение ко всем линиям микропроцессора. Часто на ПЛМ бывают заведены сигналы CLK и INT, которые никогда не подключаются к ПЗУ и ОЗУ.

Микросхемы ПЛМ наиболее трудны для анализа. Самое печальное, что невозможно дать какую-то общую рекомендацию. Слишком разнообразны как сами ПЛМ, так и схемы их подключения. Я с ними справляюсь, но начинать лучше с чего попроще. Тем более что на большей части картриджей их к счастью нет.

Разновидности картриджей

В простейшем случае в картридже установлены две микросхемы, ПЗУ программ, подключенное к шинам CPU, и ПЗУ картинок, подключенное к видеопроцессору.

В этом случае емкость ПЗУ программ на превышает 32 кб, а емкость видеоПЗУ 8 кВ.

Это самый приятный с точки зрения взлома вариант. Достаточно считать такой картридж и все проблемы решены.

Гораздо чаще попадаются картриджи, в которых кроме ПЗУ установлен регистр банков, как правило, на микросхеме SN74ALS161, бессовестно содранной с нашей 1533ИЕ10.

Хотя, вообще-то это синхронный двоичный счетчик, в картриджах DENDY он используется как регистр с параллельной загрузкой. Его выходы подключаются к старшим адресам ПЗУ программ и часто видеоПЗУ. Входы регистра могут быть подключены как к шине данных, так и к адресной шине CPU. С точки зрения анализа и взлома особой разницы нет, но при считывании появляются некоторые нюансы. Запись в регистр производится по любому адресу в диапазоне 8000h - FFFFh.

Кроме регистра могут присутствовать микросхемы мелкой логики, функции которых несложны и легко определяется после разрисовки схемы картриджа. Банки ПЗУ программ обычно имеют емкость 32 кб или 16 кб. Это зависит от того, заведена ли на ПЗУ адресная шина A14. Банки видеоПЗУ обычно имеют емкость 8 кб.

Иногда в картридже на месте видеоПЗУ установлено ОЗУ емкостью 8 кб и в одном случае я обнаружил ОЗУ в 32 кб.

К CPU ОЗУ в принципе подключить можно и вроде бы даже целесообразно, но таких картриджей я не видел. Если они и есть, то очень редки.

И наконец последняя и самая сложная разновидность.

Я имею в виду картриджи содержащие ПЛМ, (это не совсем точное название, т.к. функции заказных микросхем в картриджах DENDY посложнее простых логических матриц, но как-то же их нужно обозвать!).

Эти картриджи определяются с первого взгляда, и каждый раз как я с ними сталкиваюсь, у меня портится настроение. Практически невозможно считать такой картридж с первого раза. Приходится после считывания одного банка путем анализа программы записанной в нем определять условия для переключения на следующий. Поэтому, чтобы достоверно считать такой картридж приходится делать до 8, а иногда и более попыток. Кроме того, иногда для переключения банков формируется прерывание, а сама процедура переключения находится в подпрограмме обработки прерывания. Если две предыдущих разновидности функциональны и их структура логически оправдана, то в третьей отчетливо просматривается стремление засекретить картридж.

Совсем незначительное изменение программы, причем в сторону ее упрощения, позволило бы сделать картридж по второму варианту, с простым регистром банков.

Задача упрощается примитивностью микропроцессора и тем, что все "секретные" программы размещены в ПЗУ.

Если бы DENDY была собрана на i8080 или, хотя бы, Z-80, с их богатыми и изощренными системами команд, пришлось бы куда труднее. Хотя уже во всем мире признано, что российские, а также украинские и польские хакеры, несомненно, самые сильные. Нужда, как говорится, научит…

Есть еще одна особенность, которой картриджи отличаются друг от друга. Речь идёт о линии A10' (контакт разъема A18).

В большинстве картриджей эта линия соединена с A10 (контакт B23). При этом встроенное видеоОЗУ занимает адреса видеопроцессора 2000h-27FFh и отображается, вследствие неполной адресации, на адреса 2800h- 2FFFh, 3000h-37FFh и 3800h-3FFFh.

Линия A10' представляет собой адресный вход A10 встроенного видеоОЗУ и иногда подключается к шинам A11 (B24) или A12 (B25) видеопроцессора. При этом меняется включение встроенного видеоОЗУ в адресное пространство видеопроцессора. Подробно рассматривать эти изменения нет особого смысла. Однако в некоторых случаях вариант подключения линии A10' нужно иметь в виду.

Считывание картриджей

Для того чтобы считать картридж, нужно в первую очередь определить его тип и структуру. Кроме того, нужно иметь компьютер с дополнительным устройством типа программатора РПЗУ. Тип компьютера особого значения не имеет. Очень хорош ZX-SPECTRUM, но годится даже какая-нибудь "Микроша" или IBM PC/AT-486. Очень желательно иметь дисковод.

Картридж первого типа, не имеющий регистра банков, считывается очень просто. Достаточно через переходник, изготовленный из разъема дохлой DENDY или слота IBM-XT, подключить его к устройству считывания и считать.

Считывание картриджей второго типа осложняется необходимостью переключать банки ПЗУ. Если банки переключаются по шине данных, то перед считыванием нужно записать по одному из адресов 8000h-FFFFh нужное слово. Если же переключение происходит по шине адреса (обычно задействованы младшие адреса), то слово может быть любым, но задействованные разряды адреса следует установить соответственно банку.

Есть одна тонкость. Дело в том, что во время записи регистр банков ПЗУ картриджа включается на чтение. Если байт, который Вы подаете на шину данных картриджа, не совпадет с байтом, который записан по этому адресу в ПЗУ, произойдет так называемый "конфликт на шине данных".

Если Ваш программатор имеет достаточно мощные выходы, то он конечно "передавит" довольно хилое ПЗУ картриджа. Но в этом случае трудно поручиться за сохранность картриджа. А если программатор достаточно "деликатный", регистр банков переключится в состояние, определяемое байтом из ПЗУ.

Единственный выход из этого положения заключается в том, что записываемый байт должен быть равен байту из ПЗУ. В случае переключения банков по шине адреса нужно вначале считать байт по требуемому адресу и записывать именно его. Если же переключение идет по шине данных, то вначале нужно найти в ПЗУ адрес, по которому содержится требуемый байт, и записать этот байт именно по этому адресу. Такой байт непременно найдется, потому что эта проблема стояла в своё время и перед программистом, готовившим картридж.

Причем имейте в виду, что переключение банков видеоПЗУ также происходит при управлении по шинам CPU.

Если картридж содержит ОЗУ, то его считывать, разумеется, нет смысла.

Наибольшие трудности появляются при попытках считать картридж с ПЛМ. Если Ваша квалификация достаточно высока, то приведенной выше информации Вам будет достаточно. Дать какую-то общую рекомендацию невозможно. Каждый картридж такого типа требует индивидуального подхода.

Анализ программ

После получения копии картриджа на дискете или кассете, можно приступить к ее анализу. Для этого желательно иметь кроссмонитор, работающий в кодах DENDY. Такой кроссмонитор написан мною для ZX-SPECTRUM с системой TR-DOS. Можно использовать также компьютер, система команд которого совпадает с DENDYевской, например, COMMODORE или ПРАВЕЦ-2.

Методика анализа зависит от типа картриджа, количества игр на нем и поставленной цели. Картриджи первого типа, имеющие всего один банк, обычно содержат одну небольшую игру. При запуске сразу появляется заставка этой игры. Это наиболее простой и приятный случай. С такой игрой можно делать все, что угодно: сделать ее бессмертной, русифицировать, практически вез всякой адаптации перевести на кассету или дискету и т.д. Поскольку объём ПЗУ такого картриджа невелик, можно даже запрограммировать РПЗУ и сделать самодельный картридж.

Сложнее работать с картриджами второго типа. Первая задача - определить ведущий банк, в котором расположено начало программы. Поскольку при включении регистр банков может произвольно установиться в любое состояние, программист должен позаботиться о том, чтобы программа правильно началась из любого банка. Поэтому из любого ведомого банка есть переход в ведущий. Но здесь есть тонкость. Дело в том, что программа, переключив банк, продолжит свое выполнение уже в другом. Решается эта задача двумя различными способами. Иногда во всех банках кроме ведущего по одним и тем адресам располагается фрагмент типа

```
LOOP      LDA  #03h      ; номер  ведущего банка  03h
          STA  LOOP+1    ; запись по адресу, в котором
                         ; уже  содержится   байт  03h
                         ; сопровождается   установкой
                         ; банка  03h
```

И после этого фрагмента идет либо совершенно другая программа, либо просто "мусор", которого, кстати, в китайских картриджах более чем достаточно. Иногда даже обрывки ассемблерного текста попадаются.

А в ведущем банке есть продолжение этой программы. Дальше идет инициализация, установка стека и переменных и выход в меню.

Этот способ прост, но имеет существенный недостаток. Для его использования

нужно, чтобы во всех банках было свободное место на одних и тех же адресах, и поскольку авторы программ не подозревали, что фирма "СГИБРЕР" запихнет их программы в картриджи, они не озаботились созданием удобств для этой процедуры. Поэтому чаще используется другой вариант.

Процедура включения ведущего банка переносится из ПЗУ картриджа в ОЗУ, например, по адресу 0400h и управление передается в ОЗУ. В разных банках она может располагаться в разных адресах, но переносится в одно и то же место ОЗУ. Из ОЗУ включается ведущий банк и, после этого, дается безусловный переход с индексной адресацией на вектор START.

Как правило, в картридже второго типа бывает от одной сложной до 5 - 6 простых игр. Иногда говорят о 20, 30, даже 100 и более игр, но это неверно. Все эти сумасшедшие цифры получаются нехитрым мошенничеством. Берется одна и та же игра, меняются - начальные установки, чтобы сразу выйти на другой уровень или наделить героя новыми свойствами и все, готово. Синклеристы знают, что такое POKE. Так вот механизм "создания" десятков, а то и сотен "новых" игр именно таков. Если в картридже не одна игра, то их желательно разделить. Как правило, простая игра размещается в одном банке и во время своей работы банки не переключает. Это касается и банков видеопамяти. Таким образом, для разделения игр нужно определить в каком банке какая игра и найти адреса их запуска. Бывают тонкости, но в основном это так. Проще всего это сделать, если найти подпрограмму, формирующую меню. Для ее поиска можно использовать текст меню, который никогда не кодируется и легко обнаруживается любым монитором. Чего нельзя сказать о тексте игр. Часто он бывает закодирован. Правда алгоритм кодировки несложен и с ним без труда справляется даже одна из моих первых программ для раскодирования - "СЕРВИС-2", которая получила широкое распространение.

Работа по анализу программ DENDY несложная, синклеровские программы гораздо более "закручены", но очень нудная, из-за того, что пока нет системных программ такого качества, как для SPECTRUMа.

Заключение

Как Вы видимо убедились, легенда о полной закрытости и "секретности" DENDY не имеет ничего общего с реальностью. Конечно, совершенствовать DENDY, как ZX-SPECTRUM (от "Балтика" и "Ленинграда" к "ZS-SCORPION" и "ПРОФИ") невозможно. Но абсолютно ничего не мешает делать для нее дополнительные внешние устройства. Подключение даже кассетного магнитофона уже откроет путь для творчества. Спектрумисты дружно презирают DENDY. А собственно говоря почему? Графика у DENDY лучше, а звук не хуже чем у музпроцессора SPECTRUMа. Не потому ли, что "зуб неймет"?

Теперь, когда открывается возможность самим писать программы для DENDY, не пора ли повернуться к ней лицом и рассмотреть как следует? Ведь достаточно написать для DENDY буквально десяток системных программ, которые можно будет загружать с внешнего носителя или даже разместить в ПЗУ и SPECTRUM лишится своего решающего преимущества - открытости. Помню, когда я впервые столкнулся со SPECTRUMом, то воспринял его как неприличный анекдот на компьютерную тему. Какая - то 8-битная игрушка с мизерной памятью и несуразной консолью. Оценил его я позже и сейчас считаю, что это одна из самых удачных машин. DENDY я тоже занялся с некоторым предубеждением, а когда познакомился с ее процессором, вообще чуть не бросил. Но сейчас мое мнение уже начало меняться. Не так уж она и плоха. Во всяком случае, попробовать стоит.

Кто со мной?

Серов, 26.03.2006. Издательство "Запрещённая литература". Книгу перевёл в электронный вид Константин Жуков (player@serov.info). PDF version by NUK.